UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/712,618	11/12/2003	Edward T. Grochowski	42P15758	9270

8791 7590 11/03/2006

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
PETRANEK, JACOB ANDREW	
ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 11/03/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/712,618	GROCHOWSKI ET AL.	
	Examiner	Art Unit	
	Jacob Petranek	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *06 October 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-61* is/are pending in the application.

4a) Of the above claim(s) *56-61* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-55* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *06 October 2006* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-55 are pending.

2. The office acknowledges the following papers:

Specification, claims, drawings, and arguments filed on 10/6/2006.

Withdrawn objections and rejections

3. The specification objections have been withdrawn.

4. The drawing objections have been withdrawn due to amendment.

5. The 35 USC § 112 second paragraph rejection for claim 15 has been withdrawn.

6. The 35 USC § 112 second paragraph rejection for claims 24-30 have been withdrawn due to amendment.

New Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claims 1-8 and 14-42 are rejected under 35 U.S.C. §103(a) as being unpatentable over Bui (U.S. 2002/0056024), in view of Panwar et al. (U.S. 5,941,977).

9. As per claim 1:

Bui disclosed an apparatus comprising:

A register stack engine to trigger memory operations in support of register

windows (Bui: Paragraphs 14 and 15)(If there are sufficient registers, the RSE will spill registers out to the backing store.);

The register stack engine further to generate one or more micro-operations to perform a register window operation (Bui: Figure 1, paragraph 20)(The spill/fill operations are done by load/store instructions generated by the RSE.); and

Bui failed to teach a scheduler to schedule the one or more micro-operations for execution and wherein the scheduler is to concurrently consider the one or more micro-operations as well as other micro-operations in an out-of-order scheduling scheme.

However, Panwar disclosed a scheduler to schedule the one or more micro-operations for execution (Panwar: Figure 2 element 206, column 6 lines 19-50)(Bui: Paragraphs 26-30)(Panwar disclosed an instruction scheduler for scheduling instructions in a window-based processor. When combined with Bui in the eager mode, instructions are scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition.);

Wherein the scheduler is to concurrently consider the one or more micro-operations as well as other micro-operations in an out-of-order scheduling scheme (Panwar: Figure 2 element 206, column 6 lines 19-50)(Bui: Paragraphs 26-30)(Panwar disclosed an instruction scheduler for scheduling instructions in a window-based processor. When combined with Bui in the eager mode, instructions are scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the

processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents of the register stack in an eager mode of operations results in scheduling these operations along with other operations that the processor executes.).

Bui inherently has a way of scheduling micro-operations to execute within the pipeline, but does not specifically teach a scheduler. Using a scheduler to schedule operations out-of-order is a well-known way of sending instructions to functional units. The advantage of using a scheduler is that it will be able to detect when instructions are ready to execute and then send them onto the appropriate execution units. The advantage of using a scheduler for scheduling instructions would have motivated one of ordinary skill in the art to implement a scheduler in Bui for micro-operations. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement a scheduler for the advantage of being able to send instructions to appropriate execution units when they are ready.

10. As per claim 2:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

The register stack engine is further to insert the one or more micro-operations into an execution pipeline (Bui: Figure 3, paragraphs 26-27)(The RSE inserts store operations into the pipeline as the register pressure builds.).

11. As per claim 3:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

The register window operation is a spill operation (Bui: Paragraphs 14-15).

12. As per claim 4:

Bui and Panwar disclosed the apparatus of claim 3, wherein:

The one or more micro-operations include a store micro-operation (Bui: Paragraph 20).

13. As per claim 5:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

The register window operation is a fill operation (Bui: Paragraphs 14-15).

14. As per claim 6:

Bui and Panwar disclosed the apparatus of claim 5, wherein the one or more micro-operations include a load micro-operation (Bui: Paragraph 20).

15. As per claim 7:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

The register stack engine is further to generate the micro-operations directly (Bui: Paragraphs 20 and 26-27)(Micro-operations are instructions that perform an operation on the processor. The RSE has the ability to save registers to the backing store and thus inherently has the ability to generate micro-operations to perform the store operations.).

Bui and Panwar failed to teach the register stack engine is further to generate the micro-operations indirectly, via a micro-op generator.

However, it would have been obvious to one of ordinary skill in the art at the time of the invention to move the functionality of the RSE producing micro-ops to another functional unit. In addition, according to "In re Japikse" (181 F.2d 1019, 86 USPQ 70

(CCPA 1950)), shifting the location of parts doesn't give patentability over prior art.

16. As per claim 8:

Bui and Panwar disclosed the apparatus of claim 2, wherein:

The register stack engine is further to insert the one or more micro-operations into the execution pipeline directly (Bui: Paragraphs 20 and 26-27)(Micro-operations are instructions that perform an operation on the processor. The RSE has the ability to save registers to the backing store and thus inherently has the ability to generate micro-operations to perform the store operations.).

Bui and Panwar failed to teach the register stack engine is further to insert the one or more micro-operations into the execution pipeline indirectly, via a micro-op generator.

However, it would have been obvious to one of ordinary skill in the art at the time of the invention to move the functionality of the RSE producing micro-ops to another functional unit. It also would have been obvious to one of ordinary skill in the art that this functional unit that now produces micro-ops would also insert the micro-ops into the processor for execution. In addition, according to "In re Japikse" (181 F.2d 1019, 86 USPQ 70 (CCPA 1950)), shifting the location of parts doesn't give patentability over prior art.

17. As per claim 14:

Bui and Panwar disclosed the apparatus of claim 1, further comprising:

A renamer to rename one or more logical registers indicated in the one or more micro-operations (Panwar: Figure 2 element 204, column 5 lines 66-67 continued to

column 6 lines 1-18)(The combination result in the register stack operations in eager mode being renamed, along with other operations executing within the processor.).

18. As per claim 15:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

Each of the micro-operations is of a format that includes a single explicit destination operand and two explicit source operands (Bui: Paragraphs 19 and 20)(It's obvious to one of ordinary skill in the art at the time of the invention that the spill/fill operation could have three registers for storing the physical register to move, the pointer to the backing store, and a displacement value to state where to store the physical register value within the backing store.).

19. As per claim 16:

Bui disclosed a system comprising:

A memory to store an instruction, the memory including a backing store to store one or more spilled values (Bui: Figure 1 element 20, paragraph 17); and

Wherein the processor includes a register stack engine to generate, responsive to the instruction, one or more micro-operations to cause a register stack operation (Bui: Paragraph 20)(The spill/fill operations are done by load/store instructions generated by the RSE.).

Bui failed to teach a processor coupled to the memory, a scheduler; and wherein the scheduler is to perform out-of-order scheduling for a set of micro-operations, wherein the set of micro-operations includes the one or more micro-operations to cause a register stack operation as well as other micro-operations.

However, Panwar disclosed a processor coupled to the memory (Panwar: Figure 1 element 102, column 4 lines 45-60).

A scheduler (Panwar: Figure 2 element 206, column 6 lines 19-50)(Bui: Paragraphs 26-30)(Panwar disclosed an instruction scheduler for scheduling instructions in a window-based processor. When combined with Bui in the eager mode, instructions are scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition.);

Wherein the scheduler is to perform out-of-order scheduling for a set of micro-operations, wherein the set of micro-operations includes the one or more micro-operations to cause a register stack operation as well as other micro-operations (Panwar: Figure 2 element 206, column 6 lines 19-50)(Bui: Paragraphs 26-30)(Panwar disclosed an instruction scheduler for scheduling instructions in a window-based processor. When combined with Bui in the eager mode, instructions are scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents of the register stack in an eager mode of operations results in scheduling these operations along with other operations that the processor executes.).

Bui disclosed a method of spilling and filling register windows in a register window environment, but failed to disclose a processor that the method would be used

in. Panwar disclosed a processor that uses register windows. The advantage of using the method of Bui is that register windows can be dynamically spilled and filled with normal processing functions and thus the latency of these spill and fill operations can be overlapped with useful program work (Bui: Paragraph 15). The advantage of dynamically spilling/filling register windows would have motivated one to use this method on a register windowed processor, such as Panwar's. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to use the register spilling/filling method on the processor of Panwar for the advantage of hiding the latency of register window operations with useful work.

20. As per claim 17:

Bui and Panwar disclosed the system of claim 16, wherein:

The memory is a DRAM (Official notice is given that the memory could have been a DRAM.).

21. As per claim 18:

Bui and Panwar disclosed the system of claim 16, wherein:

The processor further includes an architectural renamer to rename registers to support register windowing (Bui: Paragraph 15)(The instruction operands are renamed so that they point to the correct register window.).

22. As per claim 19:

Bui and Panwar disclosed the system of claim 16, wherein:

The processor further includes an out-of-order rename unit to map logical registers to physical registers in order to increase parallelism (Panwar: Figure 2 element

204, column 5 lines 66-67 continued to column 6 lines 1-18).

23. As per claim 20:

Claim 20 essentially recites the same limitations of claim 3. Therefore, claim 20 is rejected for the same reasons as claim 3

24. As per claim 21:

Claim 21 essentially recites the same limitations of claim 5. Therefore, claim 21 is rejected for the same reasons as claim 5

25. As per claim 22:

Claim 22 essentially recites the same limitations of claim 11. Therefore, claim 22 is rejected for the same reasons as claim 11.

26. As per claim 23:

Bui and Panwar disclosed the system of claim 22, wherein:

The scheduler is to consider the set of micro-operations for out-of-order scheduling such that the other micro-operations and the one or more micro-operations are to be scheduled in an intermingled fashion (Panwar: Figure 2 element 206, column 6 lines 19-50)(Bui: Paragraphs 26-30)(Panwar disclosed an instruction scheduler for scheduling instructions in a window-based processor. When combined with Bui in the eager mode, instructions scheduled issued to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents

of the register stack in an eager mode of operations results in scheduling these operations along with other operations that the processor executes. Since the register stack operations and other operations are scheduled together in eager mode, they are then scheduled in an intermingled fashion.).

27. As per claim 24:

Bui disclosed a method comprising:

Performing an architectural rename stage for an instruction, in order to support register windowing (Bui: Paragraph 15)(The instruction operands are renamed so that they point to the correct register window.).

Bui failed to teach performing an out-of-order rename stage for each of the one or more micro-operations generated for the instruction.

However, Panwar disclosed performing an out-of-order rename stage for each of the one or more micro-operations generated for the instruction (Panwar: Figure 2 element 204, column 5 lines 66-67 continued to column 6 lines 1-18)(Bui: Paragraphs 26-30)(Panwar disclosed a rename stage for executing instruction out-of-order in a window-based processor. When combined with Bui in the eager mode, instructions are renamed and scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents of the register stack

in an eager mode of operations results in scheduling these operations along with other operations that the processor executes.).

Performing register renaming to support out-of-order execution is a well-known method of increasing performance in processors. Letting independent instructions to execute out-of-order and finish in-order increases the performance. One of ordinary skill in the art would have been motivated by the performance increase to implement out-of-order execution. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement out-of-order execution for the advantages of increased performance.

28. As per claim 25:

Bui and Panwar disclosed the method of claim 24 wherein:

The instruction is a procedure call instruction to invoke a new procedure (Bui: Paragraphs 24 and 25); and

Performing an architectural rename stage further comprises renaming physical register operands for a current procedure such that output registers for the current procedure are identified as input registers for the new procedure (Bui: Paragraph 15)(The instruction operands are renamed so that they point to the correct register window. The output and input operands are inherently both renamed to correctly point to the register needed to be accessed.).

29. As per claim 26:

Bui and Panwar disclosed the method of claim 24 wherein:

Performing an architectural rename stage further comprises renaming a first

input register to a predetermined physical register number (Bui: Paragraph 15)(It's inherent that the register will be renamed to a predetermined physical register number so that the renamed register will be correctly mapped to a register window.).

30. As per claim 27:

Bui and Panwar disclosed the method of claim 24, further comprising:

Generating one or more micro-operations to implement the instruction (Bui: Figure 1, paragraph 20)(The spill/fill operations are done by load/store instructions generated by the RSE.).

31. As per claim 28:

Bui and Panwar disclosed the method of claim 27 wherein:

Generating one or more micro-operations further comprises generating a micro-op to perform a desired memory operation (Bui: Figure 1, paragraph 20)(The spill/fill operations are done by load/store instructions generated by the RSE.).

32. As per claim 29:

Bui and Panwar disclosed the method of claim 27 wherein:

Generating one or more micro-operations further comprises generating a micro-op to perform an arithmetic operation associated with a register stack engine ("RSE") operation (Bui: Paragraphs 19 and 21)(An arithmetic operation is inherent when using the backing store pointer or the backing load pointer for calculating the correct address to which the data will be stored.).

33. As per claim 30:

Bui and Panwar disclosed the method of claim 27 wherein:

Generating one or more micro-operations further comprises generating a micro-op to perform a bit manipulation operation associated with a register stack engine ("RSE") operation (Bui: Paragraph 21)(Bit manipulation is performed on the implicit status register.).

34. As per claim 31:

Bui and Panwar disclosed the method of claim 24 wherein:

Performing an out-of-order rename stage further comprises mapping an architectural register to a physical rename register in order to minimize data dependencies (Panwar: Figure 2 element 204, column 5 lines 66-67 continued to column 6 lines 1-18).

35. As per claim 32:

Bui disclosed a method, comprising:

Generating one or more micro-operations to perform a register stack engine ("RSE") operation (Bui: Paragraph 20)(The spill/fill operations are done by load/store instructions generated by the RSE.); and

Inserting the one or more micro-operations into an execution pipeline (Bui: Paragraphs 26-27)(The RSE inserts store operations into the pipeline as the register pressure builds.);

Wherein the RSE operation is to support register windowing (Bui: Paragraphs 14-15).

Bui failed to teach where the pipeline includes an out-of-order rename stage for each of the one or more micro-operations.

However, Panwar disclosed where the pipeline includes an out-of-order rename stage for each of the one or more micro-operations (Panwar: Figure 2 element 204, column 5 lines 66-67 continued to column 6 lines 1-18)(Bui: Paragraphs 26-30)(Panwar disclosed a rename stage for executing instruction out-of-order in a window-based processor. When combined with Bui in the eager mode, instructions are renamed and scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents of the register stack in an eager mode of operations results in scheduling these operations along with other operations that the processor executes.).

Performing register renaming to support out-of-order execution is a well-known method of increasing performance in processors. Letting independent instructions to execute out-of-order and finish in-order increases the performance. One of ordinary skill in the art would have been motivated by the performance increase to implement out-of-order execution. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement out-of-order execution for the advantages of increased performance.

36. As per claim 33:

Claim 33 essentially recites the same limitations of claims 3 and 4. Therefore, claim 33 is rejected for the same reasons as claims 3 and 4.

37. As per claim 34:

Bui and Panwar disclosed the method of claim 33, wherein:

Generating a store micro-operation further comprises generating a store micro-operation to store data associated with the spill operation to a backing store in a memory (Bui: Paragraph 20).

38. As per claim 35:

Bui and Panwar disclosed the method claim of 32, wherein:

Generating one or more micro-operations further comprises generating a micro-operation to operate on an implicit operand (Bui: Paragraphs 19 and 21)(The backing store pointer, load pointer, and status register are all implicit operands.).

39. As per claim 36:

Bui and Panwar disclosed the method of claim 35, wherein:

Generating one or more micro-operations further comprises generating a micro-operation to perform an arithmetic operation on an implicit operand (Bui: Paragraphs 19 and 21)(An arithmetic operation is inherent when using the backing store pointer or the backing load pointer for calculating the correct address to which the data will be stored.).

40. As per claim 37:

Bui and Panwar disclosed the method of claim 35, wherein:

Generating one or more micro-operations further comprises generating a micro-operation to perform a bit-manipulation operation on an implicit operand (Bui: Paragraph 21)(Bit manipulation is performed on the implicit status register.).

41. As per claim 38:

Bui and Panwar disclosed the method of claim 35, wherein:

The implicit operand is a status bit collection register (Bui: Paragraph 21).

42. As per claim 39:

Bui and Panwar disclosed the method of claim 35, wherein:

Generating a micro-operation to operate on an implicit operand further comprises generating a micro-operation to collect a status bit into the implicit operand (Bui: Paragraph 21)(The status register is copied to the backing store as well as the data at the end of the transfer of the register window operation.).

43. As per claim 40:

Bui and Panwar disclosed the method of claim 35, wherein:

Generating a micro-operation to operate on an implicit operand further comprises generating a micro-operation to restore a status bit value from the implicit operand (Bui: Paragraph 22)(The status register is restored via the backing store upon a fill operation being finished.).

44. As per claim 41:

Claim 41 essentially recites the same limitations of claims 5 and 6. Therefore, claim 41 is rejected for the same reasons as claims 5 and 6.

45. As per claim 42:

Bui and Panwar disclosed the method of claim 41, wherein:

Generating a load micro-operation further comprises generating a load micro-operation to load data associated with the fill operation from a backing store in a memory into a register (Bui: Paragraph 20).

46. Claim 9 is rejected under 35 U.S.C. §103(a) as being unpatentable over Bui (U.S. 2002/0056024), in view of Panwar et al. (U.S. 5,941,977), further in view of Sharangpani et al. (U.S. 6,065,115).

47. As per claim 9:

Bui and Panwar disclosed the apparatus of claim 2.

Bui and Panwar failed to teach a micro-operation queue and wherein inserting the one or more micro-operations into the execution pipeline further comprises inserting the micro-operations into the micro-operation queue.

However, Sharangpani disclosed a micro-operation queue (Sharangpani: Figure 3 element 310, column 6 lines 15-29); and

Wherein inserting the one or more micro-operations into the execution pipeline further comprises inserting the micro-operations into the micro-operation queue (Sharangpani: Figure 3 element 310, column 6 lines 15-29)(The combination of Sharangpani and Bui results in the RSE operations being first put into a queue before being sent to the execution pipeline.).

The advantage of having a micro-operational queue is that it provides a storage for micro-ops that are ready to execute, but are waiting for an execution unit. The use of the storage device as a holding place for instructions would have motivated one of

ordinary skill in the art at the time of the invention to implement a micro-op queue. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the micro-op queue for the advantage of storing instructions ready to execute.

48. Claims 10-13 are rejected under 35 U.S.C. §103(a) as being unpatentable over Bui (U.S. 2002/0056024), in view of Panwar et al. (U.S. 5,941,977), further in view of Kahle et al. (U.S. 6,654,869).

49. As per claim 10:

Bui and Panwar disclosed the apparatus of claim 1, wherein:

The register window operation is associated with an implicit operand (Bui: Paragraph 19)(The backing store pointer is an implicit operand.); and

Bui and Panwar failed to teach the one or more micro-operations includes a micro-operation that indicates the implicit operand as an explicit operand.

However, Kahle disclosed the one or more micro-operations includes a micro-operation that indicates the implicit operand as an explicit operand (Kahle: Column 3 lines 25-38)(The macroinstructions contain the implied or implicit operands with the instruction and when broken into micro-operations, contain formerly implied operands as now explicit operands.).

The advantage of including explicit operands within instructions is that it facilitates faster execution of the instructions (Kahle: Column 3 lines 35-38). The advantage of executing the micro-operations faster would have motivated one of

ordinary skill in the art to implement converting implied operands to explicit operands for a micro-operation. Thus, it would have been obvious to one of ordinary skill in the art to add the functionality of converting implied operands to explicit operands for micro-operations for the advantage of quicker execution.

50. As per claim 11:

Bui, Panwar, and Kahle disclosed the apparatus of claim 10, wherein:

The implicit operand is a status bit collection register (Bui: Paragraph 21-22).

51. As per claim 12:

Bui, Panwar, and Kahle disclosed the apparatus of claim 10, wherein:

The implicit operand is a store pointer register (Bui: Paragraph 19)(The backing store pointer is an implicit operand.).

52. As per claim 13:

Bui, Panwar, and Kahle disclosed the apparatus of claim 10, wherein:

The implicit operand is a load pointer register (Bui: Paragraph 19)(The backing load pointer is an implicit operand.).

53. Claims 43-44 and 49-51 are rejected under 35 U.S.C. §103(a) as being unpatentable over Bui (U.S. 2002/0056024), in view of Panwar et al. (U.S. 5,941,977), further in view of Henry et al. (U.S. 6,587,929).

54. As per claim 43:

Bui and Panwar disclosed the method of claim 32, wherein:

The RSE operation is a spill operation (Bui: Paragraphs 14 and 15); and

Bui and Panwar failed to teach generating one or more micro-operations further comprises generating a micro-operation to assign data associated with the spill operation to one half of a double-wide data register.

However, Henry disclosed generating one or more micro-operations further comprises generating a micro-operation to assign data associated with the spill operation to one half of a double-wide data register (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(Henry disclosed a method of write combining two write instructions that write to adjacent locations. The write data is stored within the write buffer until it will eventually be written to memory. The advantage of using a register in this situation is that the instruction that will eventually write the data to memory will be able to explicitly reference the data and will be able to perform the operation faster. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention that a register could have also been used as the temporary storage of the data.).

Write combining will take two stores of data that need to be written to memory and combine them into a single memory access. The advantage of write combining is that steps such as arbitration, address, error, and completion phases can be eliminated when combining two writes into a single write operation (Henry: Column 1 lines 38-56). Additional time will also be saved from only having to perform a single memory access compared to two memory accesses. These advantages would have motivated one of ordinary skill in the art to implement write combining into the processor of Bui. Thus, it would have been obvious to one of ordinary skill in the art to implement write combining

in the register windowed processor of Bui for the advantage of increased memory access performance and improving write throughput to the memory.

55. As per claim 44:

Bui, Panwar, and Henry disclosed the method of claim 43, further comprising:

Generating one or more micro-operations to store the contents of the double-wide data register to a backing store (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The combination of Bui and Henry results in the write combining data register being written to the backing store.).

56. As per claim 49:

Bui and Panwar disclosed the method of claim 32, wherein:

The RSE operation is a fill operation (Bui: Paragraphs 14 and 15); and

Bui and Panwar failed to teach generating one or more micro-operations further comprises generating a micro-operation to obtain a double-wide data value from a backing store.

However, Henry disclosed generating one or more micro-operations further comprises generating a micro-operation to obtain a double-wide data value from a backing store (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(Henry disclosed a method of write combining two write instructions that write to adjacent locations. The write data is stored within the write buffer until it will eventually be written to memory. The advantage of using a register in this situation is that the instruction that will eventually write the data to memory will be able to explicitly reference the data and will be able to perform the operation faster. One of ordinary skill

in the art would realize that this process would also be beneficial when reading memory and two consecutive reads could be combined into one to put.the data into the register with the same benefits. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention that a register could have also been used as the temporary storage of the data.).

Write combining will take two stores of data that need to be written to memory and combine them into a single memory access. The advantage of write combining is that steps such as arbitration, address, error, and completion phases can be eliminated when combining two writes into a single write operation (Henry: Column 1 lines 38-56). Additional time will also be saved from only having to perform a single memory access compared to two memory accesses. One of ordinary skill in the art would realize that this process would also be beneficial when reading memory and two consecutive reads could be combined into one to put the data into the register with the same benefits. These advantages would have motivated one of ordinary skill in the art to implement write combining into the processor of Bui. Thus, it would have been obvious to one of ordinary skill in the art to implement write combining in the register windowed processor of Bui for the advantage of increased memory access performance and improving write throughput to the memory.

57. As per claim 50:

Bui, Panwar, and Henry disclosed the method of claim 49, further comprising:

Generating one or more micro-operations to assign one half of the double-wide data value to a general register (Henry: Figure 2 element 224, column 7 lines 63-67

continued to column 8 lines 1-11)(The temporary register holds two register values and it would have been obvious to one of ordinary skill in the art at the time of the invention that each register value needs to be loaded back into the appropriate register window.).

58. As per claim 51:

Bui, Panwar, and Henry disclosed the method of claim 49, further comprising:

Generating one or more micro-operations to assign one half of the double-wide data value to a status bit collection register (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(Bui: Paragraph 21)(The temporary register holds two register values and it would have been obvious to one of ordinary skill in the art at the time of the invention that each register value needs to be loaded back into the appropriate register within register window or the status register if all register window values have been loaded.).

59. Claims 45-48 and 52-55 are rejected under 35 U.S.C. §103(a) as being unpatentable over Bui (U.S. 2002/0056024), in view of Panwar et al. (U.S. 5,941,977), in view of Henry et al. (U.S. 6,587,929), further in view of Ross et al. (U.S. 6,314,513)

60. As per claim 45:

Bui, Panwar, and Henry disclosed the method of claim 43, wherein generating one or more micro-operations further comprises:

Bui, Panwar, and Henry failed to teach determining whether a pre-determined number of prior spill operations has been performed; if not, generating a micro-operation to assign general register data to the one half of a double-wide data register

value; and otherwise, generating a micro-operation to assign status data to the one half of the double-wide data register.

However, Ross disclosed determining whether a pre-determined number of prior spill operations has been performed (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(The collection register being at capacity means that all spills have been completed for a register window.);

If not, generating a micro-operation to assign general register data to the one half of a double-wide data register value (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The result of combining Ross with Bui and Henry is that registers within the register window are stored into the write combining register until all have been transferred, which results when the predetermined number has been met.); and

Otherwise, generating a micro-operation to assign status data to the one half of the double-wide data register (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The result of combining Ross with Bui and Henry is that registers within the register window are stored into the write combining register until all have been transferred, which results when the predetermined number has been met. Once met, the last step is to store the status register to the backing store, which is initially stored in the write combining register.).

The advantage of making a determination of if a pre-determined number of spills

or fills occur is that then the processor will know that the status information can now be spilled or filled into the backing store or status register respectfully. The advantage of knowing when the spill or fill status information would have motivated one of ordinary skill in the art to implement checking for a pre-determined number of spills or fills. Thus, it would have been obvious to one of ordinary skill in the art to implement checking for a pre-determined number of spills or fills so that the status information can be properly spilled or filled from the backing store and the status register respectively.

61. As per claim 46:

Bui, Panwar, Henry, and Ross disclosed the method of claim 45, further comprising:

If the pre-determined number of prior spill operations has not been performed, generating a micro-operation to merge a status bit into a status collection variable (Ross: Figure 14 element 2124, column 19 lines 57-67 continued to column 20 lines 1-22).

62. As per claim 47:

Bui, Panwar, Henry, and Ross disclosed the method of claim 45, further comprising:

Generating one or more additional micro-operations to perform a second spill operation (Bui: Paragraph 20);

Wherein generating the one or more additional micro-operations includes:

Generating a micro-operation to assign general register data to the other half of the double-wide data register (Henry: Figure 2 element 224, column 7

lines 63-67 continued to column 8 lines 1-11)(It would have been obvious to one of ordinary skill in the art that when putting the spill data into the write combining register that one spill operation would write into one half and the other spill operation would write into the other. Otherwise, spill data would be overwritten and never properly saved.); and

Generating a micro-operation to store the double-wide data register value to a backing store (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The combination of Bui and Henry results in the write combining data register being written to the backing store).

63. As per claim 48:

Bui, Panwar, Henry, and Ross disclosed the method of claim 47, wherein generating one or more additional micro-operations further comprises:

Generating the micro-operation to assign general register data to the other half of the double-wide data register (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(It would have been obvious to one of ordinary skill in the art that when putting the spill data into the write combining register that one spill operation would write into one half and the other spill operation would write into the other. Otherwise, spill data would be overwritten and never properly saved.) only if a predetermined number of prior spill operations has occurred (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(The collection register being at capacity means that all spills have been completed for a register window.);

Otherwise, generating a micro-operation to assign status data to the other half of

the double-wide data register (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The result of combining Ross with Bui and Henry is that registers within the register window are stored into the write combining register until all have been transferred, which results when the predetermined number has been met. Once met, the last step is to store the status register to the backing store, which is initially stored in the write combining register. It would have been obvious to one of ordinary skill in the art that when putting the status data into the write combining register that the status operation would write into the other half not currently occupied by spill data. Otherwise, spill data would be overwritten and never properly saved).

64. As per claim 52:

Bui, Panwar, and Henry disclosed the method of claim 49, wherein generating one or more micro-operations further comprises:

Bui, Panwar, and Henry failed to teach determining whether a pre-determined number of prior operations has been performed; if not, generating a micro-operation to assign one half of the double-wide data register value to a general register; and otherwise, generating a micro-operation to assign one half of the double-wide data register value to a status collection register.

However, Ross disclosed determining whether a pre-determined number of prior operations has been performed (Ross: Figure 15, element 2146, column 20 lines 57-67 continued to column 21 lines 1-37);

If not, generating a micro-operation to assign one half of the double-wide data

register value to a general register (Ross: Figure 15, element 2146, column 20 lines 57-67 continued to column 21 lines 1-37)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The result of combining Ross with Bui and Henry is that registers within the backing store are loaded into the write combining register until all have been transferred, which results when the predetermined number has been met. It's inherent that then these registers stored in the write combining register are loaded into the appropriate register within the register window.); and

Otherwise, generating a micro-operation to assign one half of the double-wide data register value to a status collection register (Ross: Figure 15, element 2146, column 20 lines 57-67 continued to column 21 lines 1-37)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(The result of combining Ross with Bui and Henry is that registers within the backing store are loaded into the write combining register until all have been transferred, which results when the predetermined number has been met. It's inherent that once all registers have been put back into the register window, then the status register is restored.);

The advantage of making a determination of if a pre-determined number of spills or fills occur is that then the processor will know that the status information can now be spilled or filled into the backing store or status register respectfully. The advantage of knowing when the spill or fill status information would have motivated one of ordinary skill in the art to implement checking for a pre-determined number of spills or fills. Thus, it would have been obvious to one of ordinary skill in the art to implement checking for a pre-determined number of spills or fills so that the status information can be properly

spilled or filled from the backing store and the status register respectively.

65. As per claim 53:

Bui, Panwar, Henry, and Ross disclosed the method of claim 52, further comprising:

If the pre-determined number of prior fill operations has not been performed, generating a micro-operation to extract a status bit from a status collection register (Ross: Figure 15 element 2144, column 20 lines 57-67 continued to column 21 lines 1-37).

66. As per claim 54:

Bui, Panwar, Henry, and Ross disclosed the method of claim 52, further comprising:

Generating one or more additional micro-operations to perform a second fill operation (Bui: Paragraph 20);

Wherein generating the one or more additional micro-operations includes:

Generating a micro-operation to assign the other half of the double-wide data register data to a general register (Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(It would have been obvious to one of ordinary skill in the art that when putting the fill data into the write combining register that one fill operation would write into one half and the other fill operation would write into the other. Otherwise, fill data would be overwritten and never properly loaded. It's inherent that both data loaded are put into the appropriate registers within the register window.);

67. As per claim 55:

Bui, Panwar, Henry, and Ross disclosed the method of claim 54, wherein generating one or more additional micro-operations further comprises:

Generating the micro-operation to assign a general register to the other half of the double-wide data register value only if a predetermined number of prior fill operations has occurred (Ross: Figure 15, element 2146, column 20 lines 57-67 continued to column 21 lines 1-37)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(It would have been obvious to one of ordinary skill in the art that when putting the fill data into the write combining register that one fill operation would write into one half and the other fill operation would write into the other. Otherwise, fill data would be overwritten and never properly loaded.);

Otherwise, generating a micro-operation to assign the other half of the double-wide data register to a status collection register (Ross: Figure 14 element 2126, column 19 lines 57-67 and column 20 lines 22-48)(Henry: Figure 2 element 224, column 7 lines 63-67 continued to column 8 lines 1-11)(It would have been obvious to one of ordinary skill in the art that when putting the status data into the write combining register that the status operation would write into the other half not currently occupied by fill data. Otherwise, fill data would be overwritten and never properly loaded. It's inherent that this status register information is loaded into the status register.).

Response to Arguments

68. The arguments presented by Applicant in the response, received on 10/6/2006 are partially considered persuasive:

69. Applicant argues "Bui failed to teach or suggest out-of order scheduling or out-of order execution of micro-operations for register window operations such as spills and fills" for claims 1 and 16.

This argument is found to be persuasive for the following reason. The examiner agrees that Bui along does not teach these limitations. However, a new ground of rejection has been given due to amendment.

70. Applicant argues "Panwar scheduler doesn't insert micro-operations for a register window micro-operation into an execution pipeline or schedule such micro-ops."

This argument is found to be persuasive for the following reason. The examiner agrees that Panwar alone doesn't teach this limitation. However, the combination of Bui and Panwar disclosed scheduling register stack operations into the pipeline.

Also, in response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

71. Applicant argues "Panwar failed to teach performing an out-of-order rename stage for each of the one or more micro-operations generated for the instruction" for claims 24 and 32.

This argument is not found to be persuasive for the following reason. Panwar disclosed a rename stage for executing instruction out-of-order in a window-based processor. When combined with Bui in the eager mode, instructions are renamed and scheduled to work ahead and save the contents of the register stack to memory in anticipation of a stack overflow condition. One of ordinary skill in the art at the time of the invention would realize that this is done while other micro-operations are executing in the processor of Bui. Thus, one of ordinary skill in the art at the time of the invention would realize that these operations to save the contents of the register stack in an eager mode of operations results in scheduling these operations along with other operations that the processor executes.

Also, in response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

Conclusion

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jacob Petranek whose telephone number is 571-272-5988. The examiner can normally be reached on M-F 8:00-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jacob Petranek
Examiner, Art Unit 2183

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100